Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 7, July 2014, pg.568 – 578

RESEARCH ARTICLE

Multi-Robot Co-ordination For Box-Pushing using Embedded Controller

Ketaki.P.Dahale[1], Vijay.D.Chaudhari[2], Kantilal.P.Rane[3]

[1] Department of E&TC, Gf's Godavari COE, North Maharashtra University, Jalgaon, India

[2] Department of E&TC, Gf's Godavari COE, North Maharashtra University, Jalgaon, India

[3] Department of E&TC, Gf's Godavari COE, North Maharashtra University, Jalgaon, India

[1] ketaki.dhl@gmail.com; [2] vinud_chaudhari@yahoo.co.in; [3] kantirane@rediffmail.com

Abstract— Multi-Robot coordination for task allocation has been widely studied topic in the literature. A robot team can accomplish a given task more quickly than a single robot can by dividing the task into sub-tasks and executing them concurrently in application domains where the tasks can be decomposed. It leads to effective coverage of a large area. The principle of territoriality in homogeneous agent groups is a physical division of space and all associated recourses in order to minimize interference and maximize synergy. Each robot is assigned a working area and robots operate only in work space allocated to them in order to reduce interference between them. Dynamic task allocation allows robots to change their behavior in response to environmental changes in order to improve overall task performance. From the studies related on division of labor mechanism in the insect society it can be found that an individual engages in task performance if environment exceed their intrinsic thresholds. The advantage of centralized approach is that they can produce globally optimal plans. In our work, we exploit some of the features of territoriality, dynamic task allocation, division of labor in insect societies and centralized systems. We propose action selection without explicit communication for multi-robot box-pushing which changes a suitable behaviour set depending on a situation for adaptation to a dynamic environment. A central controller identifies the situation and issues the appropriate control signals to adapt the suitable behaviour set to it. As a result, we find out our approach is promising for designing adaptive multi-robot box-pushing. We demonstrate these concepts using three line following mobile robots. The task is to push a given box from source location to destination location. The number of robots active at a given time depends upon the weight of the box. The boxes to be moved are of different weight. The ratio of robots working at a given time change dynamically as per the changes in environment.

Keywords— Multi-robot, Embedded Controller, box-pushing, dynamic environment, centralized co-ordination

I. INTRODUCTION

Information about final paper submission is available from the conference website. A team of robots can be effectively used for many applications such as assembly tasks, manufacturing, transporting of raw materials, finished goods, etc. (material-handling robots). Multiple robots can be used instead of a single robot to accomplish a certain task in less amount of time. Intelligent robot teams must effectively and efficiently share the workload in a way similar to the humans may share to complete the task. Robots should cooperate to effectively maximize the task performance. The cooperation can be at the level of task allocation and need not propagate to the level of task execution.

Co-ordination between robots can also be used for deciding the share of task to be executed by each robot. A key difficulty in coordinating multiple robots is deciding who does what. Thus, the task allocation mechanism is an important factor in the architectural design.A decision regarding which robot should execute a task and when can be taken so that each robot is effectively used for completion of task in least amount of time possible. Now-a-days we can find that the focus of studies related to multi-robot coordination is on task allocation and equal load sharing among the robots for overall improvement of task performance. Labor division mechanisms are found to be exploited in social insect societies that are suitable for artificial, embedded systems such as multiple mobile robot platforms.Even in human and other social groups with advanced labour division, life is organised around a series of concurrent activities. For a society to function efficiently, the number of individuals (team size) involved in these activities has to be continuously adjusted such as to satisfy its changing needs.

The process regulating team size and thus modulating labour division is called task allocation. It can be evident when centralised and embodied in a special agency like a contractor dispatching men on a working site and less visible when decentralised as with neighbours providing unsupervised help after an earthquake.Multi-robot coordination strategies assume either a centralized approach where a single agent plans for the group or a distributed approach where each robot is responsible for its own planning. The key advantage of centralized approaches is that they can produce globally optimal plans. The global information about the environment can be gathered by a central controller using a network of sensors. The controller takes the decision based on this information and then accordingly instructs the robots. Moreover, the principle of territoriality i.e. physical division of workspace has been proved to be an effective tool to minimize interference in multi-robot systems. Interference is a major stumbling block in group of robots. Also, in social insect colonies it can be found that the ratio of workers performing a certain task varies continuously. The number of workers involved in a task execution depends on the task needs and environmental changes. Not all robots need to be engaged in a task execution all the time. Which robot should participate in the task execution can depend upon the factor that which robot can execute a task in less amount of time. In homogeneous robot system this may depend upon the location of a robot from a task to be executed. The robot closest to the task may execute the task. Moreover, the robot indirectly assists the other robots in the group by sharing the workload if at all the workload increases beyond a threshold. By deciding which robots should be active while which should be idle we also contribute to power saving in multi-robot systems. In order to maximize the effectiveness of a task-achieving MRS, the robots' actions must be spatiotemporally coordinated and directed towards the achievement of a given system-level task. From a few robots performing a manipulation task, to tens of robots exploring a large indoor area , to thousands of ecosystem monitoring Nano-robots, as the number of robots in the system increases, so does the necessity and importance of coordination. Coordination is defined as \the act of regulating and combining so as to produce harmonious results". In the context of MRS, coordination involves the appropriate spatiotemporal regulation of the robots' actions such that the probability of a given task or goal being successfully achieved is maximized. Coordination in MRS can also be defined as a central controller (co-ordinator) interacting with environment and then deploying the appropriate number of robots as per the task requirements. We cannot fix the number of robots that should be involved in a task execution it the nature of task varies continuously. There may be a scenario in which the workload is such that only one robot may be required to execute the task. Whereas there may be a situation when two robots may be required to execute a task or there may exist a situation where complete team of robots may be required to complete the task. Thus, the main goal is to vary the worker ratio or the number of robots participating in completion of task dynamically as per the task needs. The task needs change continuously and the decision making has to be done as to how many robots should be active at that instant. This leads to reduction in power consumption and automation of decision making. All the robots need not be active for a task which can be easily executed by a single robot. The central controller acquires the information, analyses this data to find out task needs and then automatically coordinates the robots for task execution. Also, we automatically take into account the principle of territoriality by allocating workspace to each robot. The robot always works in its own workspace even if it decides to share the workload to reduce interference. Hence, we attempt to share the workload by minimizing interference, power consumption and time consumption.

II. RELATED WORK

Mataric, Nilsson, and Simsarin [1] studied different strategies for moving a box to a set destination. They found that a two robot team succeeded not only more often than a single robot, but also was able to accomplish the task faster. In addition, they mentioned that the robots were more effective in accomplishing the task when they communicated with each other. Clearly, a multi-robot system is a good solution for the problem of pushing a large object, and is favoured over a single robot solution. Multi-robot systems are good for several other reasons. According to Aparicio and Lima [2], multi-robot systems offer robustness and adaptability not found in single robot systems. If one agent in the system is damaged or malfunctions, the task can still be accomplished with the remaining agents. This idea can be applied to maintenance as well; agents can be serviced one at a time, which results in the system having no downtime, assuming the other robots are capable of continuing the task with the absence of their counterpart. In addition, multiple robots are able to cover more ground and can specialize in doing smaller tasks that make progress towards the completion of the larger task [2]. Therefore, multi-robot systems are more likely to reliably complete large and complex tasks. Despite the many advantages of multi-robot systems, there are a number of disadvantages that must be addressed. First of all, communication between the agents in a system is difficult computationally. According to Koes, Nourbakhsh, and Sycara [3], when multi-robot systems are making a decision, they must consider the time to travel to a location, the time to wait for other robots to arrive, and the time to complete the task. Therefore, control algorithms are very complex, as efficiency is harder to realize with more agents working on a particular problem. Radio communication adds another level of complexity to the system as transmitting circuits and communication protocols must be utilized. Therefore, while a multi-robot system may be more effective, the system requires a greater level of depth and complexity than its single-robot counterpart. During the research process, applications were discovered which directly relate to our project. One particular example involved two larger robots used to move furniture to specific locations. In a more practical example, robots could be used to move heavy materials at a construction site. It is likely that robots would help considerably with such a task and might increase efficiency, but robots are usually too expensive to be beneficial financially. McLurkin proposes an application which uses coordinated robots to explore areas that may be dangerous for humans. These robots can autonomously increase the safety of make a hazardous areas. For example, an area filled with mines can be suitable for robots that are equipped with sensors designed for mine detection. The robots, once released into a minefield, would be able to detect mines without any human interaction. Once a mine is detected, the robot could communicate the location of the mine with the other robots. The other robots, using coordination algorithms and position sensors, would be able to swarm to the mine to help dismantle or mark it. The robots may talk to each other, or they may use a main robot that gives each of them directions [4]. A simple project like coordinated robotic box-pushing serves as a stepping stone to creating a more complicated and useful system, such as robotic mine-seekers. In this section we explore two areas of previous research: centralized and decentralized approaches for multi-robot coordination and load sharing mechanism or labour division mechanisms. Many research groups [1-4] have implemented centralized approaches for multi-robot coordination. The principle advantage of such centralized approaches is that optimal plans can be produced. The leader can take into account all the relevant information conveyed by the members of the team and generate an optimal plan for the team. Many research efforts have modelled distributed systems inspired by biology [5, 6]. However, the principal drawback of distributed approaches they produce sub-optimal plans. For an object transportation task, several distributed robots systems [7-13] have been proposed but most of them need common model of the whole system or need explicit communication among robots. Tight connection does not exist among the robots and the objects. Most multi-robot systems [16-19] using centralized control need explicit communication using a medium. Even for decentralized control, some systems need explicit communication [20, 21]. Such explicit communication may be expensive and unstable depending on an environment. In contrast, a multi-robot system without explicit communication is more robust and inexpensive. It is practical that an environment changes due to a fault of a robot, introduction of new robots, or task change, etc. However, most multi-robot systems [22-29] do not have an effective mechanism to deal with a dynamic environment. To cope with the problems above, Seiji Yamada and Jun'ya Saito proposed a novel action selection method for multiple mobile robots box-pushing in a dynamic environment [30]. It deed not need explicit communication and was adaptive to a dynamic environment in which the number of robots and task difficulty change. Four situations were defined with two parameters: the existence of other robots and the task difficulty. Next it was proposed propose an action selection architecture consisting of a situation recognizer and sets of suitable behaviours to the situations. Then, careful design of the suitable behaviours for each situation was done. Task or role allocation has been extensively studied in social insects [30 - 33].The study of task allocation in social insects is particularly interesting since their labour division and its regulation are organized by surprisingly simple and robust means. Task allocation within insect colonies was considered a rigid process. The ratio of active workers must vary according to task needs and environmental changes. In research work on ant-like task allocation and recruitment in cooperative robots by Michael J. B. Krieger, Jean-Bernard Billeter & Laurent Keller good modulation of the number of individuals engaged in the two possible activities: staying in the nest (inactive) and foraging (active) was presented.

Workers were switched "on" for a task if the environment exceeded their intrinsic thresholds. When workers performed a task, stimulus levels were ratcheted down, and if not then stimulus levels were increased. The model achieved a stable equilibrium when the number of workers performing a task matched the stimulus level for the task and when individual workers maintained constant task performance probabilities.

The number of workers performing the task remained low until a set-point was reached, at which the stimulus matched the thresholds of workers with higher thresholds. At this point the colony responded immediately by increasing task performance. This response prediction is similar to that of the information centre model of recruitment (100) in which social information transfer is used as the major mechanism for transmitting a task stimulus. However, the information centre model assumes no explicit genetic effects on task performance.

The foraging for work algorithm [34,35] is (a) perform any task for which there is a need, (b) once a task is performed, continue to perform the same task, (c) if this task no longer needs to be done, stay in the nest. Pacala et al [36], building on the results with the network model, developed a set of differential-equation models to investigate the effects of social group size on task allocation. In their models, individuals receive information about task needs (a) directly from the environment, (b) through their success or failure in finding task opportunities, and (c) through random encounters with other group members.

One way of understanding division of labour is to find phenomenological rules i.e. rules that describe the behaviour without reference to mechanisms that can account for the behaviour patterns of individual workers. Second question is how do workers obtain information about task needs? Task performance is a response by the worker to the environmental information it perceives. For any given task, workers may gain information from local stimuli within the nest or through interactions with other workers or both .The third question is what internal mechanisms underlie the behavioural rules for task Performance? Both genetic and hormonal effects have been shown to be important and studies are beginning to venture into the nervous system [37].

III. DESIGN OF MULTI-ROBOT SYSTEM FOR BOX-PUSHING USING EMBEDDED CONTROLLER

We address the problem of box-pushing in a multi-robot system. The problem of dynamically changing the worker ratio as per the task needs is discussed previously. We adopt centralized approach in implementing the box-pushing problem for decision-making which can create optimal and global plans. The decision of number of active robots at an instant will be taken by the embedded controller as per the weight of the box to be pushed. The concept of task allocation using controller can be demonstrated using three line following Robots to push the box of specific weight to desired location. We can make use of IR sensors to detect the line. The number of robots required to push the box is limited by embedded controller. The global information is collected by the controller. We eliminate the drawback of obtaining information by local interactions by adopting centralized approach. Thus optimal plans can be produced from global information rather than sub-optimal solutions as in distributed system. Also, we make use of a load sharing algorithm here. We set a threshold for every robot. A robot is signalled only if the weight of the box is beyond a specified limit. A single robot can execute a task if the weight of the box is considerably low. The side robots are activated only if weight of the box increases a certain limit. The all three robots are activated where there is heavy box to be pushed. As the box push to certain limit then robots become idle. Thus, we are changing the ratio of workers or number of active robots according to the task needs or weight of box.

A. Defining situations to describe a dynamic environment

First of all, we describe a task and an environment. The task of multiple mobile robots is "Push the box up to desired location." Environment consists of limit switch placed at intervals to detect the presence of the object. First box is placed on workspace, the weight on box is read as image & according to this weight robots are activated. This input is given to the central controller, thus, robot knows about the ready task from central controller. There may be three situations persisting at a given time depending on the weight of the box. The task we address is cooperatively moving a box, large relative to the size of the robots, from some initial location to a designated goal location. In sol*vin*g this problem, we take inspiration from human behaviour commonly observed when they are carrying objects or pushing some object. If a single person is capable of pushing or carrying objects he himself will complete the task. If suddenly he encounters a heavy object or an object bigger in size the supervisor will instruct another person to help him push the object. In short the supervisor will deploy the appropriate number of workers that are actually needed to complete the task of pushing the box. Here, we have boxes of different weights to be moved from source location to destination location with the help of three line-following robots. The decision as to how many robots will be required to move the box depends on the weight of the box. Thus, the number of robots working at a time varies dynamically as per the changes in environment. Thus, the multi-robot system is adaptive to the dynamic environment. In the cooperative mobile robot domain, we formally define this problem with a set of constraints:

1. The box is large compared to the robots, and the robots can only move the box by pushing through frictional contact.
2. The pushing robots cannot in general perceive the goal due to occlusion by the box.

3. There is an obstacle-free path between the box's initial location and the goal that is wide enough for the box and robots to pass (i.e., we do not consider negotiating obstacles in a coordinated fashion, only as part of individual low-level control). We use a nomenclature for three robots as RobotA, RobotB and RobotC. Each robot has two wheels for mobility. The controller acts as a situation recognizer which recognizes the workload at a time instant with the help of a controller. Every box has a sticker attached to it. The image of this sticker is captured. The image is read, segmented and finally converted into text. The exact weight of the box is extracted from the text to get the exact weight of the box. This number is then send to the controller through the USB port. The controller reads this number and activates the required number robots to push the box.

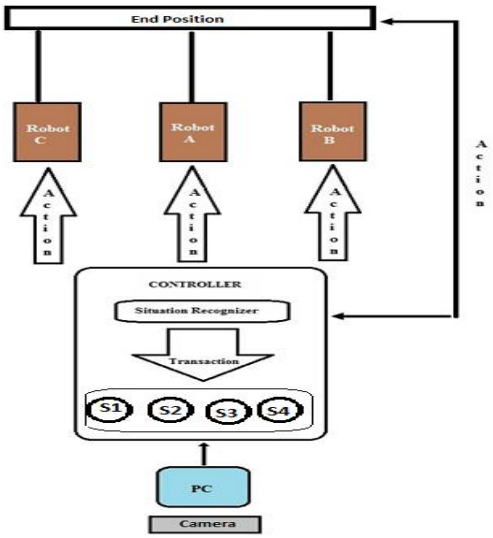

Fig. 1 A Control System for dynamic task allocation

B. Threshold Definitions

Lower Threshold (T_L): Weight of Box is in between 1-3
Moderate Threshold (T_M): Weight of Box is in between 4-6
Higher Threshold (T_H): Weight of Box is in between 7-9

We also define states of workload as T_L where workload is below threshold T_M as threshold between lower threshold and higher threshold and T_H as threshold greater than higher threshold.T_H describes heavy workload. In the following, we explain the situations and the suitable behaviours for each of them. The suitable behaviours will be concretely described by IF–THEN rule representation later. We can say that the determined situation is globally correct. In a dynamic environment, we have three robots to execute a task; all the robots need not be idle or working simultaneously. The number of active and inactive robots depends upon the availability of task and workload at a given time. We define workload in terms of the weight of a box to be pushed. We define two thresholds for defining workload i.e. lower threshold and a higher threshold. We define situation S1 as a situation with no workload and all robots are idle i.e. there is no box to be pushed. S2 can be defined as a situation with the availability of a task but a workload below lower threshold i.e. the weight of the box is such that a single robot is capable of transporting it from source location to destination location. Not more than one robot is active in this situation. S3 is defined as a state of environment where workload is moderate i.e. between lower threshold and higher threshold. Here, two robots are must to execute the task. S4 is a state where there is a heavy workload i.e. greater than a higher threshold i.e. the weight or of the box is huge enough that it will need atleast three robots to push it from source to destination. Also, we define states for robot as active and inactive. We describe the state of a RobotA with preposition 'A'. A means RobotA is active and ¬A means RobotA is inactive. Similarly, B means RobotB is active and ¬B means RobotB is inactive. C means RobotC and ¬C means RobotC is inactive. In the following, we explain the situations and the suitable behaviours for each of them. The suitable behaviours will be concretely described by IF–THEN rule representation later. We can say that the determined situation is globally correct.

1. $S1 = [\neg A \wedge \neg B \wedge \neg C]$; $W=0$
2. $S2 = [\neg A \wedge B \wedge \neg C]$; $W< T_L$
3. $S3 = [A \wedge \neg B \wedge C]$; $T_L<W<T_H$
4. $S4 = [A \wedge B \wedge C]$; $T_H < W$

C. Line Following Robot

The basic principle involved in this is it captures the line position with IR sensors mounted at front end of the robot. The block diagram of the line follower robot shows that, when the sensor senses the path, output will be 0s or 1s which are then fed to the microcontroller, and then the microcontroller decides the next move according to the program. When both the sensors are indicating low (0) then robot start moving on the black path, for white if it indicates high (1) then it moves along the path. The infrared emitter sends a continuous beam of infrared light that is reflected from the ground back to the infrared detector. Depending on the reflectiveness of the ground colour, the detector receives varying amounts of reflected infrared light. The detector is actually an infrared photo-transistor that uses infrared light to activate the transistor base. The more infrared light that gets to the detector, the more the transistor conducts. We use five IR sensors in a straight line to determine the exact position of the non-reflective line beneath the robot. By adjusting the two drive motors based on the line sensor readings, we can keep the bot cantered on the line as it drives around the track. The track must be a solid colour with no variations. A few pieces of white poster-board work best with black electrician's tape for the line. The line provides a contrast of colour that the sensors can differentiate. You can make whatever line shape you want for your bot to follow: it can be twisty, have intersections and loops, or be a simple circle. The neat thing about Linus is that he responds differently each time around the track. Because he responds to his sensors, if there is a split in the track, he might go left the first time around and right the next! It is interesting to watch the decision-making processes of this little machine.

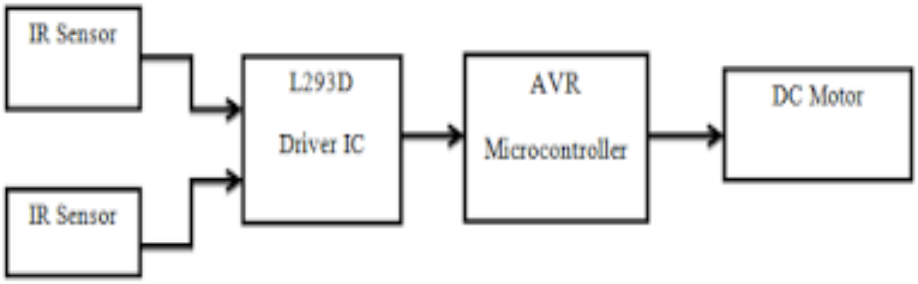

Fig. 2 Block diagram

D. The Arduino Controller

The Arduino Duemilanove is a microcontroller board based on the ATmega328 (refer annexure-I). It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 Analog inputs, a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. It's a heart of Line following robots. All the inputs from IR sensors are read by Arduino controller & according to IR input Motors are activated. The Arduino Duemilanove can be powered via the USB connection or with an external power supply. The power source is selected automatically. External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm centre-positive plug into the board's power jack. Leads from a battery can be inserted in the *GND* and *Vin* pin headers of the POWER connector. The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

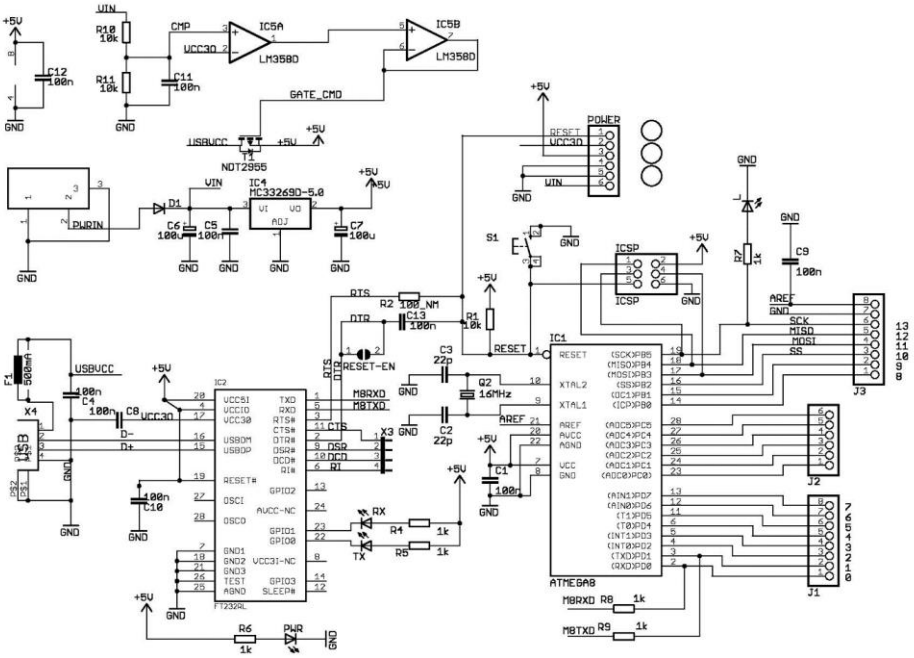

Fig. 3 Circuit diagram for line following robot

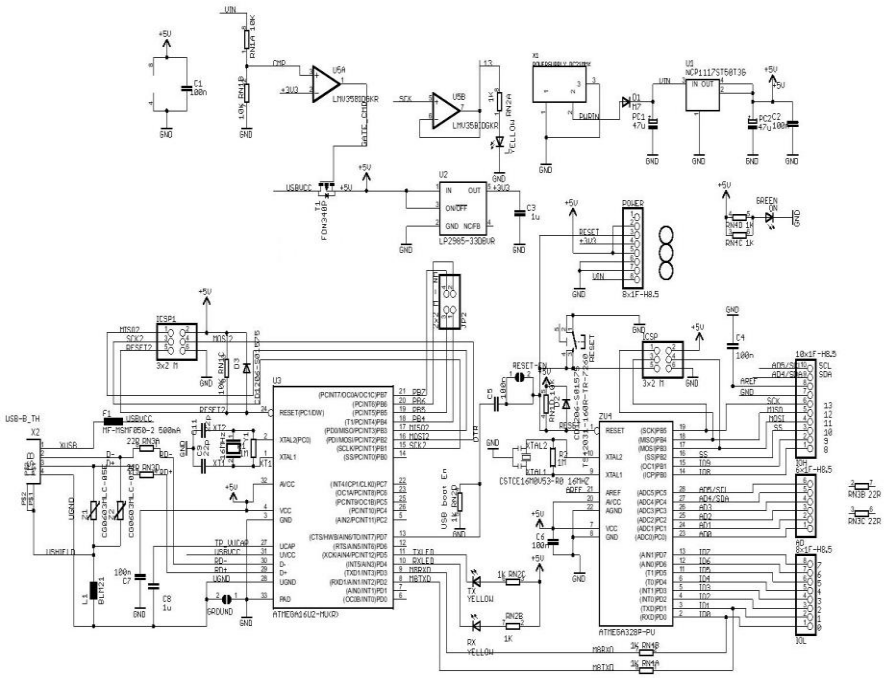

Fig. 4 Circuit diagram for Aurduino controller

IV. RESULTS

We are attempting to implement load sharing algorithm in multi-robot system using a controller for overall coordination. We are using three line following robots for demonstration. We use Arduino Atmega328 microcontroller as a controller for coordinating the robots. All three robots are used to push the box according to its weight. We prepare a model similar to an assembly line in an industry.

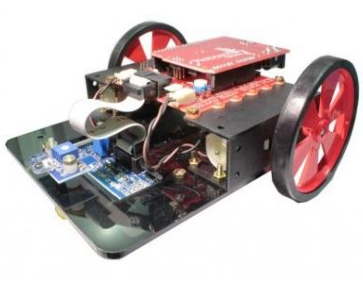

Fig. 5 Line following robot

A. Defining situations to describe a dynamic environment

First we check the action of the robots for no workload i.e. W=0.The expected result is that all the robots must remain idle. We find that all the robots remain idle in case there is no workload i.e weight of box = 0.

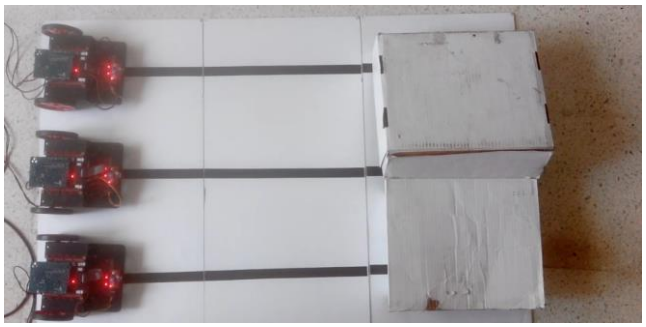

Fig. 6 No Workload

B. Results with workload below Threshold

Second we check for a situation where weight of the box is less than lower threshold .Here we define a lower threshold in between 1-3.Hence, only one robot must be active in this case where the weight of box is in between 1- 3. Now, here if we stick to the decision that only one robot should be active and it is Robot B. Robot B should execute task for weight 1-3.

Fig.7 Workload below threshold

C. Results with workload between lower threshold and upper threshold.

Next, we check the system's output for a situation with weight of the box is in between lower threshold and upper threshold. In this case only two robots must be active. We define the higher threshold to be six. Thus, we can represent the workload in this scenario if the weight of the box is between four to six. Thus, here we check the results by assigning a weight between four to six. Once Box is pushed upto certain distance and touch the limit switch, Embedded Controller stop robots A and C.

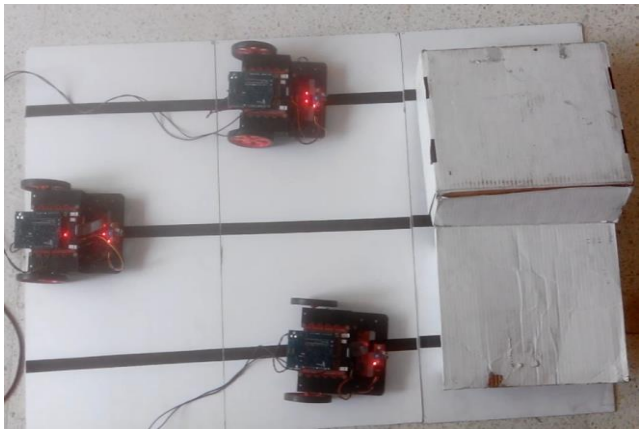

Fig. 8 Two robots completing the task

D. Results with workload greater than upper threshold.

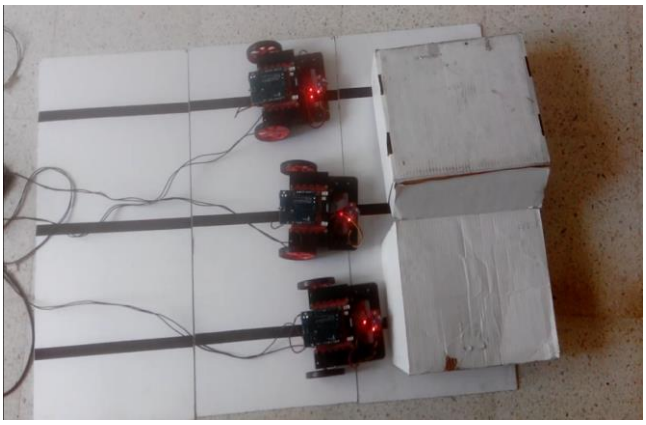

Fig. 9 Three robots pushing the box

Heavy workload may be a full workload. We have defined heavy workload as a workload greater than higher threshold. We set the higher threshold to be seven. Thus, in a scenario when weight of box is greater than seven then all the robots must be active to push the box. Thus, it can be verified that all the three robots operate at full or workload greater than higher threshold. We have defined higher threshold to be greater than five. Thus, we find that all the robots operate at full workloads.

V. CONCLUSIONS

In this work, we have designed a centralized architecture for multi-robot coordination. We have attempted to develop a flexible system that can respond dynamically to the changing environment. The number of active worker depends on the task needs. We deploy appropriate number of robots for box-pushing. The architectural design also contributes in power saving and time saving to a great extent. It can be seen that Multi-robots improve overall task performance than a single robot.

Generally distributed architecture can also be used but we generate a global plan using centralized controller. This architecture can be mainly used in industrial automation such as assembly lines and material handling. Thus, the algorithm can be implemented in other types of homogeneous robot groups also. We can also make the system configurable. The user can set the higher and lower thresholds for the system. The algorithm may be modified and implemented for large scale of applications in automation systems. In case of failure of any robot, other robots may be deployed for execution of the task for uninterrupted workflow. Task wise allocation can also be done in heterogeneous robot systems. Heterogeneous robot team means a team of robots having different architectures or configurations. The algorithm can be modified to accommodate more number of robots.

ACKNOWLEDGMENT

First of all I would like to thank my parents for their blessing. I wish to express my pleasure of acknowledging and thanks towards Mr. Vijay D. Chaudhari, for providing the technical guidelines and constructive suggestions regarding the line of this work. He encourages me all the times for doing this quality

research work in tuned with the target puts before me. I would like to place on record my deep sense of gratitude to Prof. Dr. Kantilal P. Rane, Head of Department of E&TC Engineering for his generous guidance, help and useful suggestions to improve my work.

REFERENCES

[1] Caloud, P., Choi, W., Latombe, J-C, Le Pape, C, and Yim, M., "Indoor Automation with Many Mobile Robots,"*IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp.67- 72, 1990.

[2] Jensen, R. M., Veloso, M. M., "OBDD-based Universal Planning: Specifying and Sol*Vin*g Planning Problems for Synchronized Agents in Non-Deterministic Domains," *Lecture Notes in Computer Science*, No. 1600, pp. 213248,1999.

[3] Brumitt, B. L., Stentz, A., "Dynamic Mission Planning for Multiple Mobile Robots,"*Proceedings of the IEEE International Conference on Robotics and Automation*, No. 3, pp. 2396-2401, 1996.

[4] Chevallier, D., and Payandeh, S., "On Kinematic Geometry of Multi-Agent Manipulating System Based on the Contact Force Information", *The 6th International Conference on Intelligent Autonomous Systems* (IAS-6), pp.188-195,2000.

[5] Arkin, R. C, "Cooperation without Communication: Multiagent Schema-Based Robot Navigation," *Journal of Robotic Systems*, Vol. 9, No.3, pp. 351-364, 1992.

[6] Brandt, F., Brauer, W., and WeiB, G., 'Task Assignment in Multiagent Systems based on Vickrey-type Auctioning and Leveled Commitment Contracting," Cooperative Information Agents IV, *Lecture Notes in Artificial Intelligence, Volume 1860*, pp.95-106, 2000.

[7] H. Asama, A. Matsumoto, and T. Ishida, "Design of an autonomous and distributed robot system," *in Proc. 1989 IEEE Int. Workshop Intelligent Robots Systems*, 1989, pp. 283–290.

[8] H. Kimura and Z. D. Wang, "Huge-object manipulation in space by vehicle-type robots," *JSME Int. J. Ser. C, vol. 38, no. 3*, pp. 543–551, 1995.

[9] L. Chaimowicz, T. Sugar, V. Kumar, and F. M. Campos, "An architecture for tightly coupled multi-robot cooperation," *in Proc. IEEE Int. Conf. Robotics Automation*, 2001, pp. 2992–2997.

[10] N. Miyata et al., "Cooperative transport with regrasping of torque-limited mobile robots," *in Proc. 1996 IEEE/RSJ Int. Conf. Intelligent Robots System*, 1996, pp. 304–309.

[11] J. Ota et al., "Transferring and regrasping a large object by cooperation of multiple mobile robots," *in Proc. IEEE/RSJ Int. Conf. Intelligent Robots System*, vol. 3, 1995, pp. 543–548.

[12] "In-hand dexterous manipulation of piecewise-smooth 3-D objects," *Int. J. Robot. Res*., vol. 18, no. 4, pp. 355–381, Apr. 1999.

[13] "Optimal behavior of a system of mobile robots executing a common task," *in Proc. 1994 IEEE Int. Conf. Robotics Automation*, 1994, pp. 58–63.

[14] R. C. Arkin, *Behavior Based Robotics*. Cambridge, MA: MIT Press,1998.

[15] G. Dudek, M. Jenkin, E. Milios, and D. Wilkes, "A taxonomy for multiagent robotics," *Auton. Robot.*, vol. 3, no. 4, pp. 375–397, 1996.

[16] H. Sugie et al., "Placing objects with multiple mobile robots—Mutual help using intention inference," *in Proc IEEE Int. Conf. Robot. Automat.*, 1995, pp. 2181–2186.

[17] N. Miyata et al., "Cooperative transport with regrasping of torque-limited mobile robots," *in Proc IEEE/RSJ Int. Conf. Intell. Robots Syst*., 1996, pp. 304–309.

[18] Z. Wang, E. Nakano, and T. Matsukawa, "Realizing cooperative object manipulation using multiple behavior-based robots," *in Proc. IEEE/RSJ Int. Conf. Intell Robots Syst*., 1996, pp. 310–317.

[19] K. Kosuge and T. Osumi, "Decentralized control of multiple robots handling and object," *in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*., 1996, pp. 318–323.

[20] H. Asama, A. Matsumoto, and T. Ishida, "Design of an autonomous and distributed robot system," *in Proc. IEEE/RSJ Int. Workshop Intell.Robots Syst*., 1989, pp. 283–290.

[21] L. E. Parker, "Alliance: An architecture for fault tolerant multirobot cooperation," *IEEE Trans. Robot. Automat.*, vol. 14, pp. 220–240, Apr.1998.

[22] H. Sugie et al., "Placing objects with multiple mobile robots—Mutual help using intention inference," *in Proc IEEE Int. Conf. Robot. Automat*.,1995, pp. 2181–2186.

[23] N. Miyata et al., "Cooperative transport with regrasping of torque-limited mobile robots," *in Proc IEEE/RSJ Int. Conf. Intell. Robots Syst*.,1996, pp. 304–309.

[24] Z. Wang, E. Nakano, and T. Matsukawa, "Realizing cooperative object manipulation using multiple behavior-based robots," *in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst*., 1996, pp. 310–317.

[25] D. J. Stilwell and J. S. Bay, "Toward the development of a material transport system using swarms of ant-like robots," *in Proc. IEEE Int. Conf. Robot. Automat., 1995*, pp. 766–771.

[26] R. Beckers, O. E. Holland, and J. L. Deneubourg, "From local actions to global tasks: Stigmergy and collective robotics," *in Artificial Life IV*, 1994, pp. 181–189.

[27] M. J. Mataric, M. Nilson, and K. T. Simsarian, "Cooperative multi-robot box-pushing," *in Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 1995, pp. 556–561.
[28] M. J. Mataric, "Learning in multi-robot systems," in Adaption and Learning in Multi-Agent Systems, G. Weiband and S. Sen, Eds. New York: Springer-Verlag, 1996, pp. 152–163.
[29] C. R. Kube and H. Zhang, "The use of perceptual cues in multi-robot box-pushing," *in Proc. IEEE Int. Conf. Robot. Automat.*, 1996, pp. 2085–2090.
[30] For multi-robot box-pushing*," in Proc. IEEE/RSJ Int. Conf.Intell. Robots Syst.*, 1999, pp. 1444–1449.
[31] M.D. Breed, G.E. Robinson, R.E. Page, "Division of labour during honey bee colony defense", *Behavioral Ecology and Sociobiology* 27 (1990) 395–401.
[32] Z.Y. Huang, G.E. Robinson, "Honeybee colony integration: Worker–worker interactions mediate hormonally regulated plasticity in division of labor*", Proceedings of the National Academy of Sciences USA* pp.89 (1992) 11726–11729.
[33] A. Lenoir, "Feeding behaviour in young societies of the ant Tapinoma erraticum L.: Trophallaxis and polyethism", *Insectes Society* pp.26 (1979) 19–37.
[34] Tofts C. 1993. "Algorithms for task allocation in ants (a study of temporal polyethism: theory)". *Bull. Math. Biol*. 55: 891–918
[35] Tofts C, Franks NR. 1992. "Doing the right thing—ants, honeybees and naked molerats.", *Trends Ecol. Evol*. 7:346–49
[36] Pacala SW, Gordon D, Godfray HCJ. 1996. "Effects of social group size on information transfer and task allocation", *Evol. Ecol*. 10:127–65
[37] Withers GS, Fahrbach SE, Robinson GE. 1993. "Selective neuroanatomical plasticity and division of labour in the honeybee.",*Nature* 364:238–40.

